1. The Joy of TEX

1. TEX is typesetting "language" for scientific documents. It is *incredibly* customizable and allows you define your own styles, shortcuts, etc, so that it rapidly becomes a time-saver. However, the initial stages are rough (steep learning curve!).

2. TEX has almost as many names as it has uses (AMS-TEX, LATEX, AMS-LATEX, etc) and they are all called TEX. (The one we use is actually AMS-LATEX.)

3. TEX is primarily a markup language. This means most formatting commands are carried out by "wrapping" a chunk of text with tags. For example,

   Check out *this* example!

   is typed:

   ```
   \begin{center}
     Check out \emph{this} example!
   \end{center}
   ```

   (a) Some tags, like "`center`" in this example, are called *environments* and require beginning and ending tags. These are used to place lists, tables, figures, and other chunks of non-paragraph style text into your document. There are also environments for typing mathematical things like equations, matrices, arrays, and aligned derivations of equations.

   (b) Some tags, like "`\emph`" in this example, affect only the next thing that follows them. Here, I wanted the entire word "this" to be emphasized, so I *grouped* these letters by putting it in braces "{}".

   (c) Some tags, like font commands, are like switches and affect all document text after them (within their scope, anyway).

   Knowing TEX is mostly knowing which tags to use and how.

2. What a TEX file is

TEX = ASCII + ".tex" (like HTML)

A TEX file consists of *text*, *math*, and *instructions*

Note: TEX is compiled. To view a TEX document:

1. Compile the .tex file.

2. The compiler spits out a .dvi (DeVice-Independent) file.

3. View the DVI with YAP, etc.

4. Maybe make it into PDF.

Advantages of compiling:

(1) SAVE TIME with custom keywords
(2) Document class controls layout.
(3) Automated numbering, biblio.
(4) Results are truly platform-indep.

Disadvantages of compiling:

(1) Extra step before viewing.
(2) Document class controls layout.
(3) TEX is not so robust.

3. Getting TEX running on a PC

The PCs in the lab already have software installed. Start up WinEDT, and you are ready to go.

If you want it on your home computer:

1. Install MikTex, a TEX distribution for PCs.
   Available for free at `http://www.miktex.org/`.

2. Install an editor like WinEDT.
   Shareware version available for free at
   `http://www.winedt.com/`.

Once you have the software installed,

1. Write (or load) a document.

2. Make sure the document is saved as a .tex file. Suppose you have "example.tex".

3. Click the TEX button (the bear) or press CTRL-SHIFT-X.

A window pops up and TEX compiles your document, assuming it finds no errors. Then YAP opens up "example.dvi" and you can view your handiwork. If you like, now you can press the button marked "dvi→pdf".

Look in your folder, and you will see "example.pdf", "example.log", "example.aux", "example.tex.bak".

4. Getting TEX running on a non-PC

Talk to someone else.

5. Resources

Tex is very picky and will choke on almost any mistake you make. Therefore, you will need:

1. *Math Into LaTeX* by George Grätzer.

2. *The Not So Short Introduction to LaTeX* by Tobias Oetiker & others.

3. The internet, e.g.,
   `http://math.ucr.edu/~lt/pages/tex.html`

4. `http://www.ctan.org/`
   (cf. MikTex Options → Packages) This is your source for additional TeX packages for extended functionality, etc.

5. Front-ends:
   (1) Scientific Workplace
   (2) Maple & Mathematica
   (3) MathType (for Word)
   While easy to learn/use, software like this actually reduces the power of TeX, because it does not allow for customization.

6. Examples:
   (1) `math.ucr.edu/~epearse/latex`
   All info from this talk is posted here, including the homework assignment.
   (2) `math.ucr.edu/~epearse/math_010b`
   Quizzes from a course I TAed for. This is an example of what documents look like when you DON'T use Toby's margin- overrides, as described in Section 9.
   (3) `math.ucr.edu/~epearse/math_046` or `_023`
   All course materials from courses I've taught over the summer. Includes lecture notes (with figures!), midterms, quizzes, syllabi, and final exams. Note: the exams use the `exam` document class, which has all sorts of nice features for writing multipage exams. You are welcome to steal any and all of this material.
   (4) `math.ucr.edu/~epearse/koch`
   A paper I'm currently working on. It has much more than you'd need for writing a quiz, but has lots of examples of long derivations and large formulae. Also, it shows how to split a large document into multiple files for organization, and how to write a basic bibliography.[1]

[1] For information on using the BibTeX bibliographic database for advanced bibliographies, consult *Math into LaTeX*.

6. Let's get started ...

Recall: TEX is *text*, *math*, and *instructions*.
TEX has two "modes": regular and math.

Type *text* in regular mode. Note: TEX ignores almost all whitespace in regular mode. This is very helpful for arranging your code in a readable fashion; you can indent and put extra lines between paragraphs to put space between different things, without affecting the final output in any way. Note: in text mode, TEX *will* recognize the first space or the first blank line. Otherwise, use `\par`, `\\`, `\hstr`, `\vstr`.

Type *math* in math mode. This means, enclose mathematical expressions in math delimiters. TEX ignores ALL whitespace in math mode. Note: this includes the spaces between words!

Example 1. Let $f : \mathbb{R}^2 \to \mathbb{R}$ be continuous.
This is typed:

```
Let $f:\bR^2 \to \bR$ be continuous.
```

You need $dollar sign delimiters$ on each side of the mathematical expression, or the .tex file won't compile. This is an example of "inline math".

Example 2. Suppose that we have a series

$$\sum_{n=0}^{\infty} \left(\frac{a_n}{b_n}\right)^n < \infty,$$

where $\frac{a_n}{b_n} < 1$ for all n.
This is typed:

```
Suppose that we have a series
  \[\sum_{n=0}^\infty \left(\frac{a_n}{b_n}\right)^n < \infty,\]
where $\frac{a_n}{b_n} < 1$ for all $n$.
```

Note:

(1) "displaystyle" delimiters `\[ \]`.
(2) Different $\frac{a_n}{b_n}$.
(3) Super & subscript.
(4) Braces $\{n = 0\}$.
(5) \left.
(6) n.

Inline: Suppose that we have a series $\sum_{n=0}^{\infty} \left(\frac{a_n}{b_n}\right)^n < \infty$.

Without the braces on the subscript: $\sum_n = 0^{\infty} \left(\frac{a_n}{b_n}\right)^n < \infty.$

Without the "\left" and "\right": $\sum_{n=0}^{\infty} (\frac{a_n}{b_n})^n < \infty.$

7. Errors and Debugging

Writing documents in TEX requires a lot of time for bug-hunting.

1. The log file may contain clues.
2. WinEDT will automatically jump to where *it thinks* the error is. It's usually right, or at least close.
3. Use indenting and whitespace to make your code easy to read. Start each list item on its own line, indent subenvironments, etc. This is VERY IMPORTANT.
4. Common causes of errors:
   (a) Forgetting a delimiter. (So type the closing delimiter first!) What's wrong here:
   ```
   \(\displaystyle \sum_{n=0}^\infty
     \left(\frac{a_n}{b_n}\right^n < \infty.\)
   ```
   (b) Entering math in text mode.
   `Denote the variance by \sigma^2.`
   Denote the variance by σ^2.
5. Use comments `%` to find bugs. Put a `%` in front of any lines that you think might be causing a problem. Also: use `\begin{comment}`, `\end{comment}` for multiple lines.

8. Custom Commands (The good stuff)

Example 3. Suppose that $\{f_n\}$ are uniformly continuous functions on A which converge uniformly to f.
This is typed:
`Suppose that $\{f_n\}$ are uniformly continuous`
`functions on $A$ which converge uniformly to $f$.` Or, using "text_shortcuts.sty":
`Suppose that $\{f_n\}$ are \ucn`
`\fns on $A$ which \cv \ufy to $f$.`
How to define a new command:
`\newcommand{\name}{whatever you like}`
For example: `\newcommand{\fns}{functions\xspace}`

Example 4. For Greek, see the bottom of "general.sty":
`\newcommand{\ga}{\ensuremath{\alpha}\xspace}`
The `\ensuremath` lets you type it in text mode.

Example 5. For complex conjugates:
`\newcommand{\cj}[1]{\overline{#1}}`
$\overline{z}$ =`\cj{z}` This takes 1 argument.

Example 6. For vectors:
`\newcommand{\ve}[1][x] {\ensuremath{{\mathbf{\bar #1}}}\xspace}`
This takes 1 optional argument (the default is `x`).
$\bar{\mathbf{x}}$ =“`\ve`”, or $\bar{\mathbf{y}}$ =“`\ve[y]`”.

Example 7. For limits:
`\newcommand{\limas}[2][\iy] {\xrightarrow{\hstr[1] #2 \to #1 \hstr[1]}}`
This takes two arguments, and one is optional.
$f(x) \xrightarrow{x\to\infty} \infty$ =“`f(x) \limas{x} \iy`”
$f(x) \xrightarrow{x\to 0} \infty$ =“`f(x) \limas[0]{x} \iy`”

It is generally good to keep all your custom commands in a separate file. Save them as an ASCII file “myniftystuff.sty” and include the lines
```
\NeedsTeXFormat{LaTeX2e}[1999/06/01]
\ProvidesPackage{myniftystuff}[2004/10/22 General nifty stuff]
```
Then add the line
`\usepackage{myniftystuff}`
to the preamble of your .tex file. Put the .sty file into the same folder as your .tex file, or TEX won't be able to find it. Note that “text_shortcuts.sty” is included via “general.sty”, to minimize the amount of junk in the preamble.

9. Margin Overrides

One irritating thing about TEX is that it is meant for documents which are to be published, and most books, journals, etc., have pages smaller than a standard 8.5×11.

In order to get the most out of your sheet of paper, use the override from general.sty. There is a section of code that begins with
```
%=========================================
% amsart.cls document formatting override
%   courtesy of Toby Bartels
%=========================================

\catcode `\@ 11 \setlength\footskip{75\p@}
```
and ends with
```
\catcode `\@ 12
```
Don't ask me what it means or how it works. You should be able to find the numbers in it that adjust the margins, but tamper with it at your own risk.

10. Parting words

Save often, and make backups! I like to back up all my stuff on the web site so I can download, print, view, etc, from anywhere.